2ME3 - Assignment 2

Please read this document very carefully. Follow instructions exactly. If you have any questions please post them to MS Teams or ask during office hours.

This assignment is due Nov 15th, by 11:59pm

I have created an Assignment 2 channel in Teams. If you have questions about the assignment, please post them there. Thank you.

NOTE: You may work and submit this assignment with a partner if you wish. More information on this in the Submitting and Grading section.

Purpose

The primary objective of this assignment is to assess your ability solve a proposed problem via design principles and patterns, and justify those decisions. The primary goal of this assignment is not to assess your ability to code, but rather your ability to design.

You are responsible for submitting two files:

1. A2.pdf, .txt, .docx (as long as we can read it)
2. `A2Code.zip`

See below for details on what you are responsible for completing.

Symmetric Encryption

This assignment deals with the concept of *symmetric encryption*. You are not responsible for implementing encryption schemes/algorithms, but you are responsible for knowing their properties. In general a symmetric encryption scheme has three things associated with it:

- An encryption function.
- A decryption function.
- A key. Here, a key is just an integer.

Using a scheme, we can encrypt a message using a key and the encrypt function as follows:

$$m' = \text{encrypt}(m, k)$$

In the above, m is a message we wish to encrypt, k is a key, and m' is the encrypted message. In general, it is usually the case that it is *hard* to determine m or k from m', even if the encryption scheme is known. To decrypt the message, you need to know the key that was used to encrypt it. The following statements should generally hold:

- $\text{decrypt}(\text{encrypt}(m, k), k) = m$
- If $k_1 \neq k_2$ then $\text{decrypt}(\text{encrypt}(m, k_1), k_2) \neq m$

Note, the encryption scheme is called symmetric because the key that encrypts the message is the same key that decrypts it. Perhaps the simplest (and one of the oldest) symmetric encryption schemes is the Caesar Cipher. Take a look at: `https://en.wikipedia.org/wiki/Caesar_cipher`, if you are interested. However, you should be aware that there are many more symmetric schemes which are much more complicated.

Your Tasks

For the problem described in the following section you must create a document which includes:

1. A UML class diagram of your design
2. A list of design patterns you applied and why you applied them
3. A list of design principles you feel your design enforces, and a justification as to why it does
4. An explanation of how your design is intended to be used in the overall application.

The Problem - Part 1

You have been hired to create a Spy-Network framework. The basic idea is to allow spies to communicate with other spies in the network in a secure (encrypted) manner. Moreover, to adapt to potential security threats/leaks, your network should be dynamic, in that it can change its encryption protocols (schemes/keys) at will. The following requirements should be met using proper design principles/patterns we have seen in class.

- There are three entities within the network: a home base, field bases, and spies. There may be multiple field bases and spies, but there is only one home base.
- Field bases are associated/registered with the home base, and spies are associated/registered with a single field base.
- To perform shady activities and protect the home base from liability/culpability, field bases are allowed to "go dark", in which case they can unregister from the home base. They can choose to re-register with the home-base at any point in the future.
- Spies do not have the ability to unregistered from the field base. However, they wear a device at all times such that if they die, a signal is sent to the field base to unregister them – it's like they never existed. If a "dead" spy tries to re-register with a field base the field base does not allow it.
- All entities share an encryption scheme and a key. The scheme and key are both determined by the *home base*. When the home base changes the scheme or the key, it sends the new scheme and/or key to all field bases currently in the network. In turn, the field base will send the updated information to all spies registered with it.
- Spies, field bases, and the home base all have the ability to send messages and receive messages from each other. For example, a spy can send a message to a spy, a field base can send a message to the home base, the home base can send a message to a spy, a spy can send a message to the home base, etc. When an entity sends a message they do not send it in plain text, they send an encrypted messaged using their most up-to-date scheme info. When an entity receives a message, they decrypt it using their most up-to-date scheme info and store it in some way.

The Problem - Part 2

First, fully complete Part 1. In your A2 document add a section where you describe how you would address the portion below. Also show how you would modify your class diagram to incorporate these changes. You must address these changes without modifying any of your previous code.

Your spy network is complete and working great. But after pondering which encryption scheme to use, the home base makes the observation that they can combine different encryption schemes into one. For example, consider we have two schemes: schemes A and B. Moreover, let their encryption

and decryption functions be e_A and d_A, and e_B and d_B respectively. Then for some key k one could encrypt a message as follows:

$$m' = e_B(e_A(m, k), k)$$

and then decrypt m' as follows:

$$m = d_A(d_B(m', k), k).$$

But it does not stop there, in fact, you could layer as many schemes as you wish. This is great news as it allows for even more complicated encryptions and increases the number of potential protocols in a combinatorial fashion. Add this functionality.

Submitting and Grading

This assignment will be submitted electronically via Avenue. As stated at the beginning of this document, you may work on and submit the assignment with a partner. If you choose to work with a partner, only one of you will submit the assignment. Furthermore, on the front page of your A2 document you must clearly indicate who your partner is as well as all identifying information (macid and your names as they appear on Avenue). The majority of the grades will be given to your A2 document. A rough breakdown is given below.

- 20% UML diagram
- 40% Applying correct/appropriate design patterns
- 30% Design principle justification
- 10% Explanation of how your code works, and your overall code in general

Academic Dishonesty Disclaimer

All of the work you submit must be done by you, and your work must not be submitted by someone else. Plagiarism is academic fraud and is taken very seriously. The department uses software that compares programs for evidence of similar code.

Please don't copy. The TAs and I want you to succeed and are here to help. Here are a couple of general guidelines to help you avoid plagiarism:

Never look at another assignment solution, whether it is on paper or on the computer screen. Never show another student your assignment solution. This applies to all drafts of a solution and to incomplete solutions. If you find code on the web that solves part or all of an assignment, do not use or submit any part of it! A large percentage of the academic offenses in involve students who have never met, and who just happened to find the same solution online. If you find a solution, someone else will too.